利用詞組檢索中文訴訟文書之初探

謝淳達　　鄭人豪　　劉昭麟
國立政治大學 資訊科學研究所
{ 92753008, 93753030, chaolin}@nccu.edu.tw

摘要

我們針對案情相似的訴訟文書的檢索進行研究與探討。我們以案件事實段中的詞組集合為基礎，透過這些詞組集合的比對，希望能夠更精確的找出類似於目前案件案情的過去判例，藉此幫助一般人搜尋過去的判例，並能夠從過去判例自行猜測所遇上的法律問題可能會如何被判決。此外，我們也提出了一個可以從大量文件中自動擷取可能的中文詞彙的方法，並且利用這些被擷取得到的詞彙協助我們分析判決書的事實段文字。

關鍵詞：資訊檢索、自然語言處理、法學資訊系統

1　概論

隨著科技的日益發達，以往以紙筆方式所記錄下來的資料都漸漸電子化了，不論是公司企業、政府單位或是教育單位都紛紛趕搭這班 e 化列車，司法院也不例外。近年來司法院已經把過去從民國 88 年中截至目前為止的案件判例一一鍵入電腦的資料庫，放在網路上並且提供檢索介面，方便有需要的民眾查詢。

「法學資訊系統」(Legal Information Systems) 即是利用資訊科技，協助法官和律師能夠更迅速且正確地完成他們的工作。這在國外已經有不少研究以及不少實作系統實際應用在法院的工作上。而在法律工作上常常需要對過去的法律文件做出整理以及搜尋的動作，因此有一些研究是著重在過去法律文件的搜尋，例如 Al-Kofahi 等人分別用 *k*NN、C4.5 Rules 以及 Ripper 演算法分類並且搜尋過去判例[6]，而 Ashley 和 Rissland 則是以法學專家建立的多條規則為基礎，做營業秘密法上的判例搜尋[7]。

本研究的目標即在於建立一個法律案件搜尋系統，使用者只需要將一篇新的案件事實段丟進該系統，系統會立即分析該案件事實段中的文字，並且據此到判決書資料庫中找尋過去有著相似案情的判決書，依據相似度高低排序並且將結果傳回給使用者，協助使用者快速搜尋類似案情的判例。

這個研究的動機是，對於法官而言，他們的工作就是從一個案件的事實描述中去判定某個案件中哪些人觸犯哪些法律條文，並且決定判刑刑度的輕重。然而對於少數法律規定不明的情況下，法官判刑時就會考量過去情況相似的案件的判例，作為判決的參考。一個很好的例子就是西元 2001 年的「P2P 軟體是否侵犯著作權」的議題。在以前沒有出現過 P2P 這種檔案交換模式之前，並沒有任何一個法律條文有明確規範這類案件該如何判決，一直到了國外的法院判定 Napster 這個 P2P 軟體侵犯著作權之後，國內法院將這個案件視為一個參考案例。由此可知，判例在某些案件而言有可能會是法官判決時的參考。這時如果我們的系統能自動找出可能相似的過去判例，也許可以稍微減輕法官的工作量。

對於律師而言，最重要的就是幫自己的客戶辯護。因此兩造律師在為雙方辯護的時候，往往會先去搜集與目前情況相似的過去的判例，從中找出可能對我方有利的案例，並於開庭時提出，以期法官在決策時能夠朝向較有利我方的方向思考。

判例是法院對於訴訟案件所作的確定判決的先例。在法律案件審判的過程中，對法官和律師而言，有著跟目前案件情況相似的過去判例有時是有參考價值的。因此從過去的判例中我們可以推論出目前的案件可能會如何被判決。

對於法官和律師而言，目前的司法院的裁判書查詢系統已經很夠用了，因為法官和律師都有受過專業完整的法學訓練，當他們看到一個新的案件的時候，能夠很快的判斷出新案件的類型，並且找出新案件重要的關鍵字，因此能很快的利用裁判書查詢系統，透過指定欲查詢案件的類型以及關鍵字，即可找出過去的判例。

但是查詢過去判例並不是只有法官和律師才有這類需求。一般人在遇上法律上的爭執時，有些人可能會直接花錢找專業的律師幫忙處理打官司，然而請律師打官司對某些人而言是昂貴的。如果我們的系統只需要被告將起訴書的事實段輸入，而我們的系統能夠自動分析這些文字而找出過去有著類似案情的判例，一般人就能夠利用這些判例自行判斷整個案件可能會如何判決。

雖然搜尋案情相似的過去判例對於受過專業法學訓練的律師和法官而言並不是一件難事，但是對於一般人而言，要如何精準地找出關鍵詞來搜尋過去判例是有點困難的。因此我們希望提出一個方法，協助一般人只要把起訴書的事實段描述輸入到系統中，就能自動搜尋出過去類似案件供使用者參考，讓一般人也能透過這些類似案情的判例分析並且猜測所遇到的法律案件可能會如何判決。

此研究的目的即在於幫助一般非專業法律人士的人在遇到法律問題時，只需要描述整件事情的經過，就可以透過這個系統找出過去類似情況的案例，對於一些較為簡單的案件能夠自行分析整個案件在法庭上法官可能會如何判決，進而能夠初步了解自己為何被告，以及可能觸犯了哪些法條。

一般在作資訊檢索時都是透過關鍵詞串列的比對，將每個關鍵詞視為一個單位進行相似度的計算，而我們在這裡透過將每一個句子的重要詞組視為一個單位來比對，希望能夠更精準的找出類似的案件。

本論文以電腦輔助搜尋相似案例為目標，提出了兩個方法，包括了中文詞彙自動擷取演算法，以及以詞組為基礎的相似案例搜尋演算法。

為了能更精準地分析中文判決書，我們的系統需要有足夠的知識從文章中擷取出重要的中文詞彙並且以此作為索引以及文件相似度計算的依據，而我們的中文詞彙自動擷取演算法能夠以約61%的準確率自動擷取出可能的中文詞彙。

此外，我們提出的以詞組為基礎的相似案例搜尋演算法，除了可以用來搜尋類似案例之外，也可以透過 *k*NN 的方法分類判決書的案由以及法條。在分類案由方面，跟 Liu, Chang 和 Ho[10]的方法相比，我們的方法可以用來分類更複雜的案件類型，卻維持相同的正確率。

在第 2 節中，我們會介紹目前國內外所做過的相關研究；第 3 節中說明我們蒐集資料的方法以及資料的前置處理；第 4 節介紹我們所提出的中文詞彙自動擷取演算法；第 5 節介紹我們所提出的搜尋類似判例的演算法；第 6 節將會詳細列出實驗的數據與結果；最後第 7 節則是總結並且討論我們的研究，以及提出未來可能的研究方向。

2 相關研究

2.1 法學資訊

目前已經有很多國家都設有網站提供了法律文件下載的服務。例如我國司法院的裁判書查詢系統[2]、大陸的法律法規庫、日本的裁判所網站、以及美國康乃爾大學提供的最高法院判例等法律文件的查詢系統[3]，就提供了全文檢索的搜尋介面，對於有需要查詢過去案例的律師和民眾非常的方便。

法學資訊系統的研究在國外已經行之有年了。而這些研究議題之中，如何從一堆過去的法律文件中找出與新的法律文件有高度相關者也是一個很熱門的研究主題。

Ashley 等人[7]於 1987 年建立了 HYPO 系統，這套系統是用來分析營業秘密法的案件，並且找出與目前案件有相關的過去判例。營業秘密法通常是兩家公司之間的爭執，例如商業機密竊取等等。由於營業秘密法的案件有時很難判斷孰是孰非，因此在判決上都會參考過去的判例。法律學者在營業秘密法的案例中找出一些在這類法案中常見的一些狀況(即文中的 dimension)，並且定義哪個 dimension 是對哪一方有利，例如“x's former emplyees brought x's notes, diagrams, tools to y”(對 x 有利)。HYPO 直接引用 30 個法律學者事先定義的 dimension，用來分析目前案件的事實是屬於哪些 dimension，接著就可以把與目前案件相關的過去判例找出，並且透過每個 dimension 事先定義對哪一方有利的資訊可以得知目前案件有哪些有利的 dimension 以及不利的 dimension，最後將這些關係畫出來。

Ashley[8]之後又於 1999 年提出 SMILE 系統，與 HYPO 一樣，所處理的都是營業秘密法的案件。SMILE 以句子為單位做人工標記 dimension 以及訓練語料，並且採用機器學習的技術例如 ID3 取得 Decision Tree 做為分類案件的依據。此外也採用同義詞以及一些語言學相關的資訊例如詞類、Head-Modifier relation 等語言相關的資訊，以提升分類準確率。

Liu, Chang 和 Ho[10]於 2004 年提出以過去案例來分類新案件的方法。其中為了解決過去案例過多而導致需要大量儲存空間和處理時間的問題，而將過去案例中類似的多個案件整合成單一案件來代表這類的案件。而計算兩個案件相似度的方法是分別將案件中的關鍵詞抽取出來作為有序或無序的串列，再透過公式計算得出兩個案件的相似度。

而廖[4]於 2004 年論文其中一部分是承接 Liu, Chang 和 Ho[10]的研究，也是以過去案例來分類新案件，但提出了調整詞彙權重的方法改善分類結果。廖藉由所有案件兩兩比對，依據案由的異同調整兩案件事實段的 Ordered Common Words (OCW)的詞彙權重，並且利用 Introspective Learning 的方法讓系統反覆學習以提高分類效果。

一個案件可能會因為敗訴的一方不服法官的判決，而繼續上訴，而上訴後又可能因為高等法院認為地方法院判決錯誤而駁回要求地方法院重新審理，而每一次審判結束都會有一篇判決書產生，這時會有同一個案子有許多不同的判決書的情況，這在法律上是很常碰到的問題。雖然說這些判決書都屬於同一個案件，但是這些判決書因為某些因素，有時彼此並不會記錄彼此的關連性，如果往後有法官或律師想去了解這類過去的案件的話，若無法把所有的判決書都找出來的話是無法了解整個案情的來龍去脈的。因此 Al-Kofahi 等人於 2001 年提出一個系統，幫助法官和律師將某個案件的所有過去的判決書全部搜尋出來，大幅減少蒐集資料的時間[6]。

2.2 中文詞彙自動擷取

中文文件與英文文件最明顯的差別就在於英文文件中，字與字之間都會有空白字元作為分隔，因此電腦很容易可取得文件中所有的英文字，而中文文件中雖然有標點符號區隔每個中文文句，但是一個連續的文句中間的所有字以及詞彙之間都沒有任何分隔，這使得我們沒辦法取得句子中的詞彙，因為無法得知句子中的詞彙是從哪個字開始，到哪個字結束。句子中最重要的部分往往是詞彙，若我們將一個句子中每個字分開來看是無法看出整個句子所代表的意思，而一個句子即使把中間的連接詞等功能詞都移除，只留詞彙序列我們還是可能可以猜出原本整個句子的意思。詞彙可以說是一個個概念的元素，因此在做檢索系統的時候我們也大多都是以詞彙作為索引。這種情形在法律文件中也是一樣的，因此研究中文詞彙擷取技術是我們的系統中很重要的一部份。

Sproat 與 Shih[13]於 1990 年提出以統計學常用的 Mutual Information 方法，試圖找出中文字串中每個辭彙的邊界。但是該方法的缺點是只能找出二字詞詞彙，對於所處理的語料庫中有許多三字以上詞彙的應用而言似乎不太適用。

Chien[9] 於 1997 提出了以 PAT-Tree 為基礎的中文詞彙擷取方法。該研究利用 PAT-Tree 可快速

存取資訊的特性，並且運用詞頻以及兩個詞彙的 Mutual Information 判斷一字串是否可能是有意義的詞彙，可以幫助我們快速地在一堆中文文句中找出所有的中文詞彙。

Yang 與 Li[15]受到 Chien[9]的啟發，於 2002 年提出了改進的方法。該研究主要是想嘗試解決 Chien 的方法的一些小缺點，由於 Chien 的方法所找出來的字串有不少是由有意義的辭彙前後再加上一些例如「的」、「之」這類的 function word，而這類的字串對我們而言是比較沒有意義的，因此 Yang 與 Li 嘗試解決這個問題。

Tsay 與 Wang[15]嘗試把中文詞彙歸類為群集(clusters)，藉此降低計算的複雜度。Tsay 與 Wang 並實驗使用 Rocchio 線性分類器、naïve Bayes 和 *k*NN 做為文章分類方法之效能評估。結果顯示 *k*NN 是這三種分類方法中效果最好的一個，這正好支持我們目前使用 *k*NN 作為分類機制的核心。本文研究的方法事實上跟 Tsay 與 Wang[15]所提的架構相當相似，最大的不同在於我們是以詞組作為檢索作為基礎，而 Tsay 與 Wang 是以詞作為檢索基礎。此外，我們所做的分類其實沒有一個絕對的標準，所謂「相似案情」的判例的標準答案隨著測試案例的不同而會有所改變。

3 前置步驟

我們從司法院網站上的裁判書查詢系統[2]，下載台灣各地方法院的刑事法庭中，從民國 88 年 8 月至民國 94 年 2 月之間，屬於竊盜、搶奪、強盜、贓物、傷害、恐嚇這六大類的判決書。由於實驗中必須使用到判決書的事實段文字，因此我們將所有缺少事實段的判決書刪除，並且將所擷取的判決書依照案由分類，每一類依照判決時間排序，將最新的前 20%的判決書作為 test data，而剩餘的 80%則做為 training data，統計如下：

表 1 判決書數量統計

	竊盜	搶奪	強盜	贓物	傷害	恐嚇
訓練資料	1600	1600	1600	1600	710	241
測試資料	400	400	400	400	179	60
總數	2000	2000	2000	2000	889	301

表 1 的統計數據並不能夠完全正確地反應判例的真實案由。刑事案件有許多情況都不只有單一案由，例如強盜案件往往都是犯罪者先偷竊，被人發現後不得已才強取甚至毆打被害人。然而判決書的案由僅為案件中最重要的案由，為了統計方便，上面的統計數字都是以判決書上面的案由為準。

4 詞彙之取得

4.1 取得中文的重要詞彙

為了分析判決書事實段中的文字，必須有個方法能夠幫助我們找出文字中所有有意義的詞彙。一般而言，要取得中文的重要詞彙有下列兩種方法：

1. 利用中文詞典。語言學家所建立的詞典對我們要找出文字中有意義的詞彙有相當大的幫助。著名的研究有我國的中央研究院中文詞知識庫小組建立的中央研究院平衡語料庫（Sinica Corpus）以及中華民國計算語言學會開發的中文詞知識庫及中文語法(CKIP Lexicon and Chinese Grammar) 常用中文詞的電子詞典。我們的研究是利用 HowNet[1]。這些詞典都是以人工的方法編纂並且電子化，因此電腦可以透過電子詞典取得大量的中文詞彙以及這些詞彙的詞性(動詞、名詞等)與類別(竊盜、車輛等)等有用的資訊。
2. 利用特殊的演算法，例如上述 Chien[9]所提出的從一堆文章中自動擷取出中文詞彙的方法。

上述兩種方法各有優劣。第一種方法好處是可以取得詞彙的詞性與類別等有用的資訊，但是這類的辭典只有將常見的詞彙建入，但是法律文件中有許多詞彙甚至專用術語是 HowNet 中沒有的，例如「行動電話」、「朋分」、「水果刀」等重要詞彙。第二種方法的好處是我們可以透過演算法將某些特定領域常出現的重要詞彙都擷取出來，因此可以補足上述第一個方法的缺點，然而方法二卻無法直接提供我們所有詞彙的詞性與類別。

4.2 詞彙特性

由於法律文件中有許多重要的詞彙是 HowNet 中所沒有的，因此我們採用第二種方法，利用我們提出的方法找出在法學文件中常出現的重要詞彙。

我們認為詞彙有下列兩點特性：

1. 詞彙的出現次數超過某個頻率 t_f。
2. 字數大於兩個字的詞彙，其中任兩個相鄰的字的同時出現的頻率也大於頻率 t_f（例如「行動電話」這個詞彙的出現次數若為 n 次，其中的「行動」、「動電」、「電話」的出現次數也一定大於 n 次，因為每一次出現「行動電話」都代表其中的「行動」、「動電」、「電話」也會同時出現一次）。

根據上述兩點特性，設計了以下的演算法。首先說明演算法中使用到的符號意義：

令 c_n 為文中第 n 個字，w_n 為字組$\{c_n, c_{n+1}\}$，S1 與 S2 為用來記錄詞頻的資料結構，f_n 為 w_n 在 S1 之出現次數，sep(w_n)代表在字組$\{c_n, c_{n+1}\}$兩字之間放一個假想的分隔線。t_f 為詞頻的 threshold，t_d 為相鄰兩個短詞可視為單一長詞的最大可容忍頻率差，last_freq 為記錄前一個詞彙中最小的 bi-gram 的詞頻的變數。以下是演算法的內容：

Input: 所有檔案
Output: 這些檔案中所有可能的詞彙集合

Step 1: 讀取所有檔案中的文字，將文句中所有出現過的相鄰兩個字的字組的出現次數記錄於 S1 中。若其中一個字為標點符號則忽略之。

Step 2: 第二次讀取所有檔案中的文字。

對於每一篇文章 A：

let last_freq = 0;

```
for n=0 to length(A)-1 {
if(fn>=tf) {
if(last_freq > fn*td) sep(wn);          //case1
else if(fn > last_freq * td) sep(wn-1);  //case2
else last_freq = min(last_freq , fn);    //case3
}
else {sep(wn);   last_freq = fn;}        //case4
}
```

一篇文章經過上述方法處理過後，每篇文章中會有許多因 sep()而產生的假想分隔線，我們將兩兩分隔線之間的字組視為一可能的詞彙，並且將這些詞彙的出現次數記錄於 S2。

Step 3: 將 S2 中所有詞彙以下列方法過濾：

(a) 若詞彙的第一個字為數字則移除之

(b) 若詞彙的最後一字為「之許十百千萬時日一二三四五六七八九內中外前後於」其中一字，則移除之。

Step 4: 最後程式將會視在 S2 中出現次數多達 t_f 次以上的詞彙為有意義的詞彙並且輸出。

上述演算法中，Step1 的目的是記錄任何兩個字在文件中相鄰的次數有多少，例如「手持西瓜刀、行動電話」這段文字，會將「手持」、「持西」、「西瓜」、「瓜刀」、「行動」、「動電」、「電話」這些 bi-gram 的出現次數都加一。

Step2 的目的是找出文中的詞彙以及這些詞彙的邊界。我們以字組 w_n 為單位，以 w_n 在 S1 之出現次數 f_n，作為判斷的考量。相鄰的字組若是出現的次數(頻率)相近的話，則可能是一個連續的長詞。在實作上我們以最大可容忍頻率差 t_d 界定所謂頻率相近的範圍，若是前後變化超出這頻率的範圍則加上一個分隔線。使得在兩分隔線之間的字組是頻率接近的。Step2 之中 case1 與 case2 是判斷目前位置是否為詞彙的右邊界，以及在左邊若有一個兩字以上的長詞彙，則以 last_freq 變數記錄該長詞中最低頻的 bi-gram 的詞頻(case3)，以「手持西瓜刀」這句子為例，「手持」會因為出現次數多而不被分開(case2)，「持西」因為跟「手持」相比詞頻低太多而被分開(case1)，而「西瓜」、「瓜刀」都很常出現而且兩者的詞頻差異不大，因此「西瓜刀」被視為一個單一的詞彙(case3)。

而 Step3 主要是透過事先建立的 stop word list，將符合條件的詞彙視為不可能的詞彙而刪除之。

在 Step2 中的 case1 和 case2 會遇到的問題是兩個相鄰的短詞是否可看做是一個長詞。例如「台北地方法院」這個詞彙是由「台北」和「地方法院」兩個短詞，上述的設計是兩個詞彙中若較常出現的詞彙的頻率比另一個詞彙的頻率高出一定程度 t_f 時，則不將這兩個短詞合併為單一長詞，但若有需要也可以將此設計改為若合併後的單一長詞的出現頻率高於 t_f，則該長詞與兩個短詞各累加一次。

4.3 方法分析

這個方法只能找到兩個字以上的詞彙，雖然有些單一字元也會有重要的意思，然而要讓電腦自動找出這些重要的單一字元較為困難，目前所參考的中文詞彙自動擷取研究也都只能擷取兩字以上的詞彙，因此在這個研究中我們暫時不考慮這些單一重要字元的擷取。

由於透過我們的方法所找出的詞彙並非絕對是有意義的詞彙，因此我們會以人工的方法過濾這些詞彙，留下所有有意義的詞彙。我們的方法找出來的詞彙，經過人工過濾後有 2055 個詞彙，其中有 1485 個詞彙是 HowNet 中有定義的，而 HowNet 沒有定義的詞彙有 570 個。而這些 HowNet 沒有定義的詞彙有不少在訴訟文書中算是很重要的關鍵詞，而這些 HowNet 沒有定義的詞彙佔了所有詞彙約 1/4，對於搜尋的結果仍有不小幫助，因此以我們提出的方法找出特殊領域中重要的詞彙還是有其必要性。關於此方法的 performance 議題我們將在第 6 節實驗結果討論。

5 搜尋類似判例之方法

5.1 中文斷詞方法

一個句子是由多個詞彙以及字元所組合而成，而斷詞的工作就是將一個句子斷成許多個詞彙及字元組成的序列。因此斷詞的工作必須要有詞典的幫助。在這裡我們採用前面第四節所介紹的我們提出的方法來建立訴訟文書的詞彙列表，作為斷詞的依據。

斷詞可分為長詞優先與短詞優先兩種方法。例如「行動電話」這個詞，若使用長詞優先方法將被斷成「行動電話」，而使用短詞優先的方法則將被斷成「行動」與「電話」兩個詞彙。長詞優先的斷詞方法會比短詞優先的方法更為正確，更能表示文句中想要表達的意思。因此在我們的研究中採用長詞優先的斷詞方法，並且將所有判決書的事實段作斷詞的工作。

5.2 同義詞的定義

在我們所處理的判決書中，同一個概念常常會看到以不同詞彙表達的情況。例如「竊盜」這個概念，在判決書中就可能會以「竊取」、「竊得」、「行竊」等詞彙表達。如果我們將這些詞彙視為不同的個體處理的話，可能會使得準確率降低，而且本研究又是以詞組為基礎，如果一個詞組是由兩個詞彙 w1-w2 組成，而這兩個詞彙分別有三個同義詞，那麼這個詞組的概念就有九種可能的組合，將有可能使最後的準確率降低，因此我們有必要有個機制取得詞彙的同義詞。

雖然 HowNet 有定義詞彙所屬的類別，然而 HowNet 並非是專門為法學領域而設計，因此有一些法學領域的概念並沒有被定義在 HowNet 中，例如「贓物」、「交易」、「侵入」等概念就沒有在 HowNet 中被定義；此外，HowNet 中對很多詞彙都定義了許多類別，例如「共同」這個詞彙在 HowNet 就被分類到 aValue、attachment、public、behavior、together 這些類別，但程式如何得知哪個類別才是「共同」這個詞彙在法學文件中最常見的類別呢？此外，如同之前所提過的，在我們所處理的法律文件中，有 1/4 的詞彙沒有出現在 HowNet 中，亦即我們無法透過 HowNet 查詢到這些詞彙所屬的類別。綜合上

面三點，因此我們不採用 HowNet 中對於詞彙類別的定義。

在沒有其他可供參考的資料之下，我們以人工的方式，以自己的定義[5]，將所找出來的所有詞彙一一分類，目前共有 113 個類別，編成同義詞辭典，作為程式判斷同義詞的依據。

5.3 詞組的定義

我們將詞組定義為：兩個詞彙的組合。例如一個句子中有「偷竊」與「贓車」兩個詞彙，則我們說「偷竊-贓車」是該句子的其中一個詞組。由於中文句法的關係，我們以逗號或者是句號所間隔開的字串作為一個中文句子。

一般的搜尋系統通常是利用詞彙的序列作文件相似度的計算基礎，然而我們將以詞組作為相似度計算基礎。例如我們要搜尋「偷竊信用卡」這個概念，如果有個判例中有「偽造信用卡」這個句子，如果以單一詞彙為基礎作索引的話，系統會因為兩個案件都有「信用卡」這個概念而增加相似度的計分，但是我們知道這兩個概念是截然不同的，我們希望如果過去判例中同時有「偷竊」和「信用卡」兩個概念時才增加其相似度分數。因此我們採用以詞組為基礎的搜尋方法，希望利用詞組能夠更精準地抓到文件中的文意，藉此提升相似判決書的搜尋效果。

在這裡先說明我們產生詞組的方法：

Input: 一個句子

Output: 該句子中所有的詞組配對集合 P

Step1：將句子經過斷詞，取得句子中所有的詞彙

Step2：如果詞彙被歸類在某個同義詞類別時，則以該類別的名稱代替該詞彙。

Step3：將這些詞彙或類別名稱兩兩配對，但忽略詞組中兩個詞彙的順序，並且移除兩個相同詞彙所形成的配對。最後將過濾後的所有詞組加入集合 P 中。

在我們的方法中，我們將忽略詞組中兩個詞彙的順序，亦即「砍傷-被害人」和「被害人-砍傷」這兩個詞組將會被視為相同的詞組。雖然有時詞彙的出現順序是重要的，但是由於事實段中常常會有倒裝句，例如「將被害人砍傷」這樣的句子，若是考慮順序的話，系統將會認定「將被害人砍傷」和「砍傷被害人」是不同的兩件事情，因此我們不考慮詞組中兩個詞彙的順序。

5.4 找出重要的詞組的方法

以詞組為基礎所會遇到的問題是：哪個詞彙該跟哪個詞彙配對？對於大部分的句子而言，句子經過斷詞後我們會得出兩個以上的詞彙，例如「手持客觀上足以對人之生命構成危險之水果刀一把預藏於身上」這個句子，經過長詞優先方法斷詞後會得到「手持」、「客觀上」、「生命」、「危險」、「水果刀」這些詞彙，我們該如何得知「手持」必須跟「水果刀」組成一個詞組呢？

為了解決這個問題，我們提出一個可以找出重要詞組的方法。詳細說明如下：

Input: 所有判決書的事實段

Output: 可能重要的所有詞組的集合 S

Step1：將所有事實段文字以標點符號做斷句，取得所有句子的集合 ST

Step2：過濾 ST 中所有句子，只保留符合下列兩點條件的句子，其餘則刪除：

(a) 句子長度小於 t_c 個字

(b) 句子經過斷詞後只有小於 t_w 個詞彙但至少有兩個詞彙的句子蒐集起來

Step3：利用 5.2 節提出的方法，一一找出 ST 所有剩餘的句子中的所有詞組，並且累加這些詞組的出現次數。

Step4：將 Step3 中所有出現次數超過 t_p 次的詞組加入集合 S 中。S 中的所有詞組集合即為可能重要的所有詞組。

在上述演算法的 Step2 中之所以要設定這兩個限制，是為了解決上面我們提到的不知道如何配對的問題。我們認為一個句子中所含的詞彙個數越少，我們的配對錯誤率會越低；而一個句子的字數越少，代表該句子中會因為我們詞彙資料庫的資料不夠而在斷詞時被我們忽略的詞彙越少。在後面的「實驗結果」章節中，我們會對演算法中的 t_c 和 t_w 這兩個參數分別設定不同的數值分別實驗最後的案件分類效果，實驗結果顯示 t_w 愈小會使得正確率提升；而 t_c 愈小也會使得正確率提升，但影響不大。在我們的實驗中，我們將 t_p 固定在 10。

透過上述方法而取得了重要詞組集合 S 之後，我們在找尋一個新句子重要的詞組時，只要先利用第 5.3 節提出的方法，找出該句子中所有的詞組集合 P，最後將 P 跟 S 作交集後，得到的集合就是該句子中所有可能的重要詞組。

透過上述方法，我們可以利用簡單的交集運算得出句子中可能的詞組，而不用煩惱句子中哪個詞彙該與哪個詞彙配對的問題。雖然上述的方法找出的 S 並不是每個詞組都是有意義的詞彙配對，但是根據我們的觀察，此方法可以找出大部分有意義的詞組，並且刪除大部分不可能的詞組，由「實驗結果」章節可得知，透過這個方法搜尋相似案件的效果還算不錯。

5.5 利用詞組搜尋類似案件的方法

我們將一篇判決書事實段的每個句子，以第 5.4 節所描述的方法處理過之後，可以取得每個句子中重要詞組的集合，我們將這個集合視為該案件的特徵。有了這些詞組，我們就可以計算兩個案件的相似度。令 i_1 和 i_2 為兩個案件的重要詞組集合，$u_{1,2}$ 為 i_1 和 i_2 的交集，$count(x)$ 為任一詞組集合 x 的詞組數目，則相似度公式如下：

$$sim(i_1, i_2) = \frac{count(u_{12})}{\sqrt{count(i_1) \times count(i_2)}} \qquad (1)$$

使用者輸入起訴書事實段時，系統會把使用者輸入案件以及過去所有的判例以上述相似度計算

公式計算其相似度分數，並且依照相似度分數排序，將前幾名最相似的過去判例輸出供使用者參考。

6 實驗結果

6.1 中文詞彙擷取演算法

我們將上述的「中文詞彙擷取」演算法套用在訓練用判例共 7351 篇判決書的事實段，我們採用的參數，對照於第 4.2 節中的符號，是 t_f = 30，t_d=1.5，亦即詞頻的 threshold 為 30 次，且相鄰兩個短詞可視為單一長詞的最大可容忍頻率倍數差為 1.5 倍。實驗結果如下：

表 2 中文詞彙擷取結果

程式分析	人工過濾	詞頻	正確率
前 30 名	剩 29 個	>3711	97%
前 100 名	剩 94 個	>1656	94%
前 300 名	剩 263 個	>532	87.7%
前 500 名	剩 423 個	>305	84.6%
前 1000 名	剩 796 個	>137	79.6%
前 1310 名	剩 1007 個	>100	76.9%
前 1500 名	剩 1125 個	>84	75%
前 2000 名	剩 1400 個	>57	70%
全部 3359	剩 2055 個	>30	61.2%

為了讓接下來的「相似判例搜尋」演算法搜尋出來的結果能更好，因此在這裡我們刻意將詞頻的 threshold 設定的較寬鬆，讓程式將出現次數只有 30 次的詞彙也視為可能有意義的詞彙，期望從中取得更多有意義的詞彙，如此在分析判決書事實段時才不會遺漏太多資訊。所謂有意義的詞彙，我們的作法以人工過濾認為是個詞彙的都納入詞彙庫，而對於詞性，我們沒有實際進行語法分析而是以人工會將這些"有意義的詞彙"中，挑出屬於動詞的詞彙，並且標示為動詞，因此詞彙只分為動詞和非動詞兩種詞性。在上述實驗中，程式找出的第 1310 名以後的詞彙的出現次數都低於 100 次，從表 2 中可得知出現次數高於 100 的詞彙的準確率有 76.9%以上，而詞頻大於 30 的詞彙的擷取正確率也有 61.2%。

而最後人工過濾後所得出的 2055 個詞彙中，有 1485 個詞彙在 HowNet 中有定義，因此有 570 個詞彙，約 1/4 個詞彙是透過「中文詞彙擷取」演算法而得，對於文件分析而言，這 1/4 的詞彙將對分析結果有不小幫助。

6.2 搜尋相似判例

我們從測試用判例中，亂數取出六大類案件各 5 篇，亦即共 30 篇測試資料，輸入至系統中，由系統從 training data 中分別找出最相似的前 5 篇判例，並且以人工閱讀之方式評估之。

表 3 相似判例搜尋結果

	0 篇	1 篇	2 篇	3 篇	4 篇	5 篇
竊盜	2	0	0	0	2	1
搶奪	1	0	0	0	2	2
強盜	1	1	1	1	0	1
贓物	0	2	0	1	1	1
傷害	1	0	0	0	2	2
恐嚇	1	2	0	1	0	1

在此說明上表中欄位的定義，在上表中的第二列第二行的數字 2 代表著所查詢的 5 篇新的竊盜案件中，有其中 2 篇新案件（即 2 次 query）搜尋到的結果都只有 0 篇判例與之相似；而最後一列最後一行的數字 1 代表著所查詢的 5 篇新的恐嚇案件中，有其中 1 篇新案件（即 1 次 query）搜尋到的結果中 5 篇判例都與之相似。

上表中值得注意的是，第二行的數字代表著系統傳回的結果都與新案件不相似的次數。造成搜尋結果錯誤的原因，除了我們在整個方法上應該有可以進步的空間外，在某些判例的描述中，常常會有「於附表所示時、地，以附表所示之方法犯罪」這類的字樣，這造成了搜尋上的困難；另外如同前面所述，許多案件都可能牽涉到兩種以上罪行，例如竊盜之後被發現結果發生扭打造成傷害就變成強盜，此例中如果系統因詞彙庫不夠完整而使得無法分析出其中造成轉變的要素（被發現、造成扭打等詞），就會錯誤分析新案件，而使得搜尋結果變得不正確。

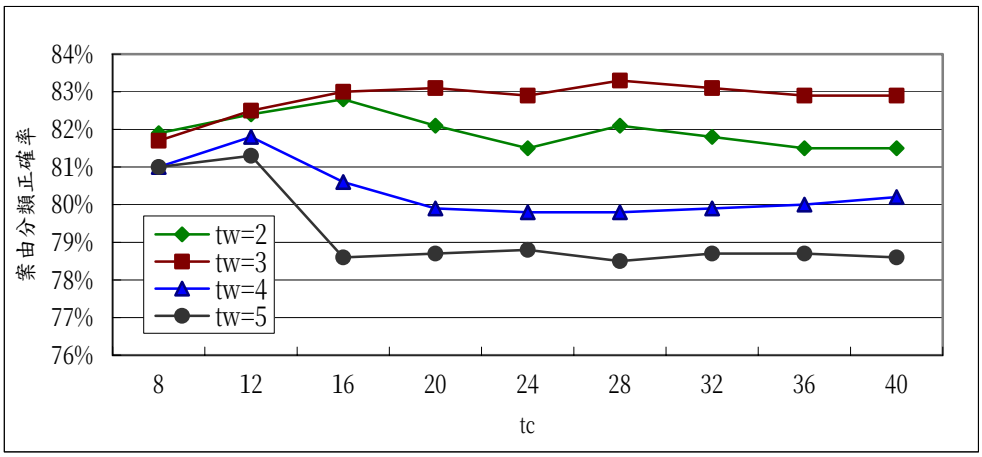

圖 1 各種 t_w 和 t_c 組合的案由分類結果

6.3 以相似判例猜測輸入案件之案由

我們將一個案件輸入到系統後，會傳回前 X 個系統認為與該案件最相似的過去判例，而每一篇判決書的開頭都有標註該案件的案由。既然搜尋到的案件被認為是跟輸入案件最為類似的，那麼我們可以大膽假設這些搜尋到的案件的大部分的案由都會跟輸入案件的案由相同。因此我們可以透過搜尋到的這些最相似的判決書的案由，以最常出現的案由來猜測輸入案件最可能的案由。

我們分別將所有測試用判例作為輸入案件輸入到系統，將參數 X 固定為 10，並且將第 5.4 節中提及的兩個變數 t_w 和 t_c 分別設為不同的數值做實驗，實驗結果如圖 1（前一頁）。

由圖 1 可觀察出，平均而言 t_w 數值設為 2 或 3 時分類效果最好，這個結果支持前面第 5.4 節中曾經提過的概念，亦即若一個句子中有太多詞彙時，由於我們不知道哪個詞彙該跟哪個詞彙配對，很容易造成配對錯誤的情形，而使得搜尋結果降低。而從圖 1 中也可得知，在 t_c 大於 16 時，t_c 對於分類效果的影響就不再顯著，反而是 t_w 參數影響較大。

由上面的實驗數據得知，在 t_w=3 和 t_c=28 時分類效果最好，有 83.3%的正確率，我們再把 t_w 和 t_c 參數固定為 3 和 28，並且將 X 設為不同的數量分別實驗，得到的結果如下：

表 4　t_w=3 和 t_c=28 時各種 X 下的分類結果

X	1	3	5	10	20	30
Accuracy(%)	76.9	81.4	82.4	83.3	83.1	82.3

由表 4 可得知，系統搜尋出的前 10 篇的判例案由與輸入案件相同的機率是最大的。下表是 X=10、t_w=3 和 t_c=28 時的案由猜測統計結果（令 C1~C6 依序代表竊盜、搶奪、強盜、贓物、傷害、恐嚇此六大案由，P 代表 precision，R 代表 recall，F 是 P 和 R 的調和平均；表 5 中第 3 行第 2 列的數值 38 代表所有案由為 C2 的案件中，被系統判斷為 C1 案由的有 38 篇）：

由表 5 可得知，竊盜、搶奪、強盜與恐嚇四種類型案件有時容易彼此誤判，這是因為這些類別的案件其相似度很大，例如竊盜案件只要多出傷害脅迫被害人的行為就變成強盜案件、而搶奪和強盜案件也因為嫌犯往往會恐嚇被害人而使得案情跟單純的恐嚇案件相似度較高，因此在判斷上較不容易，而使得誤判的情形較多。至於傷害案件跟其他類型的案件差異較大，因此判斷的正確率也較高。

表 5　t_w=3、t_c=28 和 X=10 時案由猜測結果

	C1	C2	C3	C4	C5	C6
C1	382	38	81	15	0	18
C2	6	347	72	2	0	2
C3	0	5	211	0	0	1
C4	12	5	6	379	0	2
C5	0	4	27	1	178	4
C6	0	1	3	3	0	34
P	71.5%	80.9%	97.2%	93.8%	83.2%	82.9%
R	95.5%	86.8%	52.7%	94.8%	100%	55.7%
F	81.8%	83.7%	68.4%	94.3%	90.8%	66.7%

Liu, Chang 和 Ho[10] 於 2004 年曾提出以詞彙序列為基礎的案由分類方法。在該論文中嘗試結合各種方法實驗，其中跟此研究比較相似的是代號為 E10 的實驗結果，該實驗是以 HowNet 作斷詞的工作，並且以斷詞後的詞彙序列作為計算案件相似度的基礎，與本研究的差別在於我們有使用中文詞彙擷取演算法而非使用 HowNet，有定義同義詞，並且以詞組的集合的方式作為案件相似度的計算基礎。這些方法的實驗結果整理如下 ([10] 中有許多正確率計算方法，在此我們採用該論文中所指的 WR 的計算公式，這同時也是表 6 中我們計算新系統正確率的方法)：

表 6　與其他案由分類方法之比較結果

實驗	本研究	E10
正確率	83.3%	87%

表 6 的數據乍看之下，本研究的方法似乎跟 [10] 的方法效果上有些微差距，但是由於[10] 中所處理的判決書範圍為簡易法庭中的簡易判決書，而簡易法庭是專門審理案情較為單純的案件，因此簡易判決書中的事實段內容通常字數比較少，而且所使用的詞彙多大同小異。此外，[10] 研究中所處理的 12 種案件類型中，彼此的差異比我們目前所採用的六類案由相對地要大，例如其中的妨害風化罪、賭博罪、傷害罪、違反毒品防制條例等案件，通常是沒有共通的重要詞彙，因此在判斷相似度時較不容易出錯。而在本研究所處理的案件都是刑事案件，其事實段較為複雜，而字數大部分都較簡易判決書的事實段為多，且我們處理的案件類型中，竊盜、搶奪、強盜、贓物共同的特徵很多，例如有些竊盜案件中多加一兩個元素進去，例如竊賊毆打被害人之類的，就變成一個強盜案件了，因此在判斷上較為困難。

雖然本研究所處理的案件類型較為複雜，但正確率卻與[10]的方法差異不大，因此我們的方法還是有其存在的價值。

7　結論與未來展望

在此研究中，我們提出了中文詞彙自動擷取的方法，自動從所有判決書事實段中擷取出所有的詞彙。我們也提出了找出重要詞組的方法，沒有嘗試判斷一個句子中的動詞和受詞組的問題，而把詞彙區分為動詞與非動詞，並且利用以詞組為基礎的概念，實作一套可搜尋過去相似案件的系統。我們希望這個系統能夠協助一般人快速且有效地搜尋出與他們所遇到之案件相似的過去判例，讓使用者能夠透過這些相似判例來推敲他們可能將會面臨到什麼樣的法律判決。

關於我們在第 4 節提出的中文詞彙自動擷取方法，我們認為還有改進其擷取正確率的空間。在

文章中一個詞彙若是左右兩邊都剛好是另一個詞彙或標點符號，也就是一個詞彙的邊界的話，那麼我們應該有更大的信心認為該詞彙是個完整的、有意義的詞彙。例如「…，搶奪被害人的財物，…」這段文字，在我們目前的方法之中，會將「搶奪」、「被害人」、「財物」視為可能詞彙的候選人，並且將這些字串的出現次數累加。由於「搶奪」被標點符號以及「被害人」緊緊連接，因此我們可以對「搶奪」這個詞彙有更大的信心說他是一個完整的詞彙，而透過某種機制給予較高的分數。而「被害人」是左邊緊接著一個詞彙，右邊連接著不被我們視為詞彙的字元，而「財物」左邊連接著不被我們視為詞彙的字元，右邊則緊接著一個標點符號，這兩個詞彙分別只有一側緊接著一個詞彙或標點符號，而另一側則為一個字元，因此我們給予較低的分數。

至於我們在第 5.4 節提出的尋找可能重要詞組的方法，我們認為如果能夠對每個詞組計算其在六大類別案件中的分布情形做統計，並且利用這些分布的變異數來進一步區分重要與不重要的詞組，應該有助於提升搜尋結果的正確性。例如「竊取-項鍊」這個詞彙集中分布在竊盜和強盜這兩類案件的話，此詞彙的分布變異數必定較高，因此我們認定該詞為較重要的；若是某個詞組較平均地分布在各類案件的話，則我們認定該詞組較不重要而予以刪除。我們希望透過這個方式能進一步刪除較不重要的詞組，進而提升正確性。

致謝

本文同時承蒙本研討會及中華民國人工智慧學會所主辦之第十屆人工智慧與應用研討會之評審委員的建議，增刪部分文字以提高論文可讀性。由於我們主觀地堅持這一篇論文是一篇中文版論文，因此沒有遵照本研討會評審委員之建議加入英文版的摘要，謹此致歉。感謝國科會研究補助案 NSC-93-2213-E-004-004 及 94-2213-E-004-008 之資助，使參與計畫人員得以完成工作。

參考文獻

[1] HowNet 電子字典 http://www.keenage.com/

[2] 台灣司法院判決書查詢系統 http://nwjirs.judicial.gov.tw/FJUD/index.htm

[3] 國外判決書查詢系統：

大陸法律法規庫 http://search.law.com.cn:8080/

日本裁判所 http://www.courts.go.jp/index.htm

美國最高法院判例等法律文件查詢 http://www.law.cornell.edu/index.html

[4] 廖鼎銘，*觸犯多款法條之賭博與竊盜案件的法院文書的分類與分析*，碩士論文，國立政治大學，台北，台灣，2004。

[5] 謝淳達，*利用詞組檢索中文訴訟文書之研究*，碩士論文，國立政治大學，台北，台灣，2005。

[6] K. Al-Kofahi, A. Tyrrell, A. Vachher and P. Jackson, "A machine learning approach to prior case retrieval", *Proc. of the 8th Int'l Conf. on Artificial Intelligence and Law*, pp. 88-93, 2001.

[7] K. D. Ashley and E. L. Rissland, "But, see, accord: Generating blue book citations in HYPO", *Proc. of the 1st Int'l Conf. on Artificial Intelligence and Law*, pp. 67-74, 1987.

[8] S. Bruninghaus and K. D. Ashley, "Toward adding knowledge to learning algorithms for indexing legal cases", *Proc. of the 7th Int'l Conf. on Artificial Intelligence and Law*, pp. 9-17, 1999.

[9] L.-F. Chien, "PAT-tree-based keyword extraction for Chinese information retrieval", *Proc. of the 20th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 50-58, 1997.

[10] C.-L. Liu, C.-T. Chang and J.-H. Ho, "Case instance generation and refinement for case-based criminal summary judgments in Chinese", *J. of Information Science and Engineering*, Vol. 20, no. 4, pp. 783-800, 2004.

[11] M. F. Moens and R. Angheluta, "Concept extraction from legal cases: The use of a statistic of coincidence", *Proc. of the 9th Int'l Conf. on Artificial Intelligence and Law*, pp. 142-146, 2003.

[12] J. Osborn and L. Sterling, "JUSTICE: A judicial search tool using intelligent concept extraction", *Proc. of the 7th Int'l Conf. on Artificial Intelligence and Law*, pp. 173-181, 1999.

[13] R. Sproat and C. Shih, "A statistical method for finding word boundaries in Chinese text", *Computer Processing of Chinese and Oriental Languages*, Vol. 4, no. 4, pp. 336-351, 1990.

[14] P. Thompson, "Automatic categorization of case law", *Proc, of the 8th Int'l Conf. on Artificial Intelligence and Law*, pp. 70-77, 2001.

[15] J.-J. Tsay and J.-D. Wang, "Design and evaluation of approaches to automatic Chinese text categorization", *Int'l J. of Computational Linguistics and Chinese Language Processing*, Vol. 5, no. 2, pp. 43-58, 2000.

[16] W. Yang and X. Li, "Chinese keyword extraction based on max-duplicated strings of the documents", *Proc. of the 25th Annual Int'l ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 439-440, 2002.